Highlights

BIOSTRAT 2.0 : Design and construction of the stratospheric cubesat

Thomas Rimbot, Gabriela Bergiel, Sara Gancarczyk, Wiktoria Wiejak, Agata Kołodziejczyk

BIOSTRAT 2.0 : Design and construction of the stratospheric cubesat

Thomas Rimbot[a], Gabriela Bergiel[b], Sara Gancarczyk[b], Wiktoria Wiejak[b], Agata Kołodziejczyk[c]

[a]*European Organization for Nuclear Research CERN, Meyrin, Switzerland*
[b]*AGH University of Science and Technology, Krakow, Poland*
[c]*Analog Astronaut Training Center, Krakow, Poland*

Abstract

The manuscript provides practical guidelines on prototyping and implementing a project based on the construction of a module for conducting biological experiments in the stratosphere. The module will be lifted using High Altitude Balloons. The BioStrat project aims to design a CubeSat for the same purpose. To address the associated challenges at the prototyping stage, the manuscript opted to adhere to CubeSat standards. It describes the design of the enclosure, computational components based on ESP32, and the power module.

Keywords: High Altitude Balloons, construction, CAD, ESP32, ADXL534

1. Introduction

Goal: prepare easy-to-assumable and easy-to-maintain universal platform for experiments performed in stratospheric environment.

2. Materials and Methods

This section describes the participants, materials and procedures involved when conducting this experiment during the YGGDRASIL mission. It also contains the safety procedures which have to be followed before each experiment to ensure maximum protection to the analog astronauts.

2.1. Assumptions

This manuscript is meant to be a user manual and assembly instruction for a universal platform in cubesta standard, ment for preforming biological experiments in stratosphere. The payload will be raised to the stratosphere via meteorological balloon. This instruction covers only assembly of main component which can be modified to suit best for particular experiment.

2.2. Materials

- 1 x ESP32 platform
- 1 x 20mmX80mm prototyping board
- 1 x 60mmX80 mm prototyping board
- 1 x ADXL345 accelerometer
- 1 x BMP280m termometer and pressure sensor
- 1 x SD card module
- 1 x MP1584 step-down power step down converter
- 1 x battery case for 4 AA batteries

- 4 x Energizer Lion batteries
- JST-XH 2,54 female connectors
- 2 x Female socket 1x15 raster for goldpins 2,54mm

2.3. Safety and installation

As project involves working with low voltage and soldering, basic principals with working with such dangers should be implemented. Ensure to wear appropriate protective gear such as heat-resistant gloves and safety glasses to prevent burns and eye injuries. Always work in a well-ventilated area to avoid inhaling fumes from the soldering process. Make sure to unplug the soldering station when not in use and handle the equipment with care to avoid electric shocks or short circuits. Additionally, be cautious of hot soldering irons and avoid touching the heated elements directly to prevent burns.

3. Assembley

3.1. About the design

The main case was designd to fit into CubeSat standard.

CubeSat standards are specifications for small, cubic satellites, ensuring compatibility between the missions. The primary standard, the CubeSat Design Specification, developed by Cal Poly and Stanford University, sets the basic parameters like size, mass, and deployment methods.

This basic design is meant to fit in restrictions for 1U CubeSat (not including outer passive isolation), especially in case of dimensions and mass, which are:

- Max. mass: 1.33 kg
- Max. demotions: 100mm x 100mm x 100mm

This case can fits 3 shelves (counting from the bottom):

- Shelf 1: power unit of max. size 90mm x 90mm x 20 mm
- Shelf 2: main controller unit of max. size 90mm x 90mm x 20 mm
- Shelf 3: experiment unit of max. size 90mm x 90mm x 20mm

3.2. Printing parts

3D printing is a fast prototyping method. It is additive, which means that the material is added not taken away (in contast to machinig). For this step any 3D printer with work field of dimensions bigger than 10 cm X 10cm X 10cm is needed.

The components that will be printed are:

- Case - base (fig. 1)
- Case - sides (fig. 2)
- Shelves - can be adjusted to mission objectives, (example on fig. 3)

Simplified flow of preparing and printing a prototype is:

1. Make 3D model id modeling software of your choice, export it to .stl format
2. Open .stl file in slicer of your choice, most popular one are Prusa Sicer, more beginner friendly is Cura Slicer.
3. Position parts so they do not overlap each other

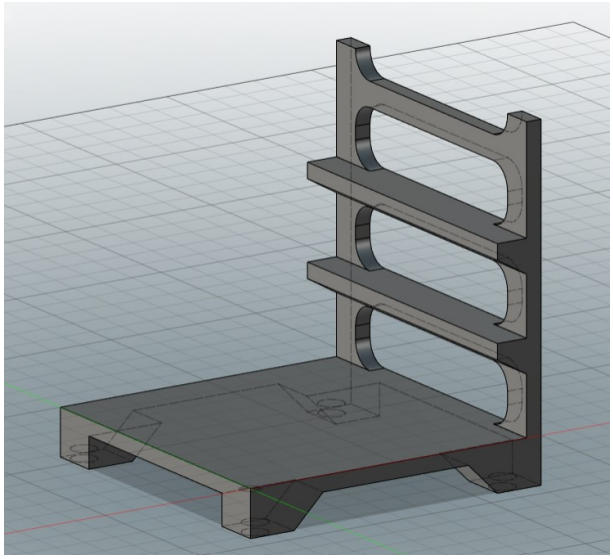


Figure 1: Model of the base of a case

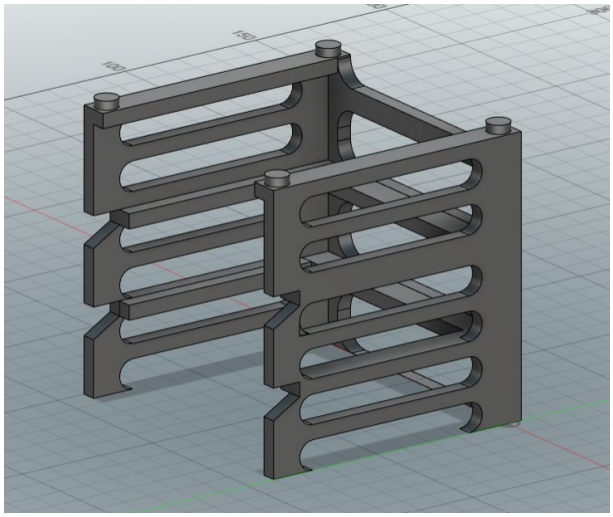


Figure 2: Model of the sides of a case

4. Generate Gcode, save it on SD card or USB (depends what is needed for your 3D printer, in habitat Prusa 3D Printer needs SD card in FAT32 format). Gcode is a set of instructions for 3D printer how to move nozzle and bed via stepper motores.

Components were designed using Fushion360 software, but any other can be used as well. Very user friendly alternative for beginners is **TinkerCad**. For gcode preparation Prusa slicer is used. Any other slicer can be used as well.

All presented parts are meant to be printed without supports. It is recommended to print base and sides together if workspace of used 3d printer is big enough. **Before slicing be sure they are positioned as on fig. 4** . It can be done by rotate icon on the left of the screen. Simmilary with shalves:

Then press Slice (in the right bottom of a screen). View similar to this should appear:
Save your print on SD card, put SD card in a printer - be sure lock switch is in the right position. Full print of base

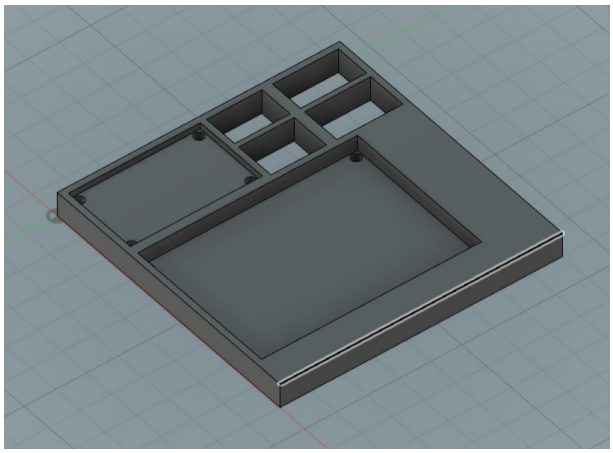


Figure 3: Model of main controller unit shelf

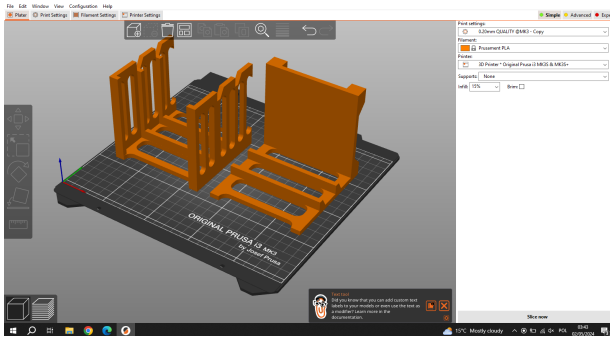


Figure 4: Short Title- Long Description

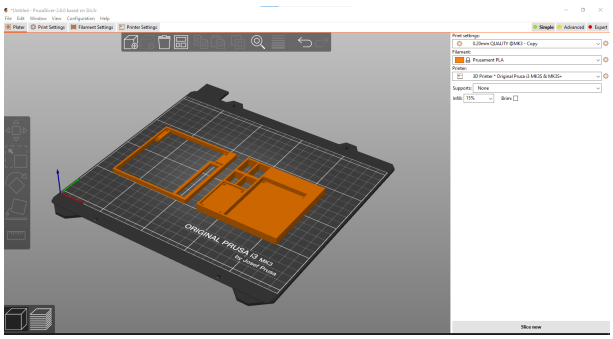


Figure 5: Short Title- Long Description

and sides of case would take about 13h. Full print of one shelf would take about 1h - 1.5h.

3.3. Preparing new version of shelves for experiment

You are welcome to prepare your shalves and setups for scpecific experiments. To make them compatibile with case:

- Their maximum **outer dimensions** have to be **90 mm X 90 mm X 25mm** or **90 mm X 90 mm X 35mm**

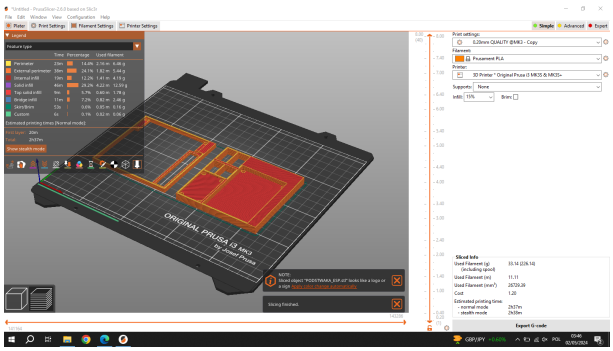


Figure 6: Short Title- Long Description

3.4. Assembly of controller unit

3.4.1. About the design

Main controller used during Biostrat missions is ESP32 board with ESP-WROOM-32 on 30-pin module. It was chosen due to elasticity, small demotions and immunity to bit-loss in low temperatures. It was mounted in universal board. During the mission due to lack of equipment it was soldered directly to the prototyping board. It is recommended to use a gold-pin socket instead (will be described belowe).

3.4.2. Assebmly

The main platform used to manufacture main controller unit will be double-sided prototyping board in size 50mm x 70 mm. It was chosen due to its flexibility if design or routing changes is needed and due to low price. Skills required for this part are:

- basic electronics knowledge (eg. Ohms Law, reading the schematics)
- basic soldering skills

Tools required for this part are:

- soldering station
- sharp, small pincers
- wires in multiple colours
- multi-meter

The basic unit has 4 components:

- ESP32 module
- ADXL345 acceleration sensor
- BMP280 pressure and temperature sensor
- SD card reader module

The connections are shown on a schematic on fig. 11:
In the design:

1. Sensors: ADXL345 and BMP280 are connected to the ESP32 via I2C interface
2. Sensors are powered not by ESP32 3.3V but from 5V source from power unit.
3. SCL and SDA lines are connected to 4.7K pull-up resistor.
4. SD card is connected to the ESP32 via SPI interface

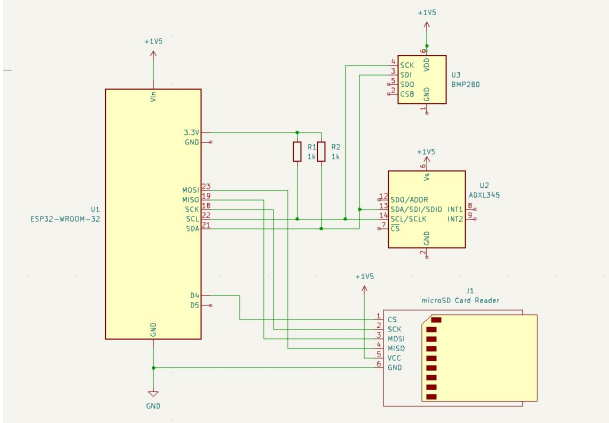


Figure 7: Schematics of connections of base unit.

3.4.3. Practical tips

During soldering parts together, please follow the tips:

- The connections responsible for power distribution will be routed only on a bottom of prototyping board.
- The connections from GPIO to neighbouring pin will be routed on the bottom of prototyping board.
- The signals connections will be routed on the top of prototyping board.

After every connection check if there is no short circuit! Use multi meter set on icon marked with yellow circle on 8

Example of routing the power lines on the back of prototyping board are shown on fig. 9

Procedure of assembly of the the main controller unit:

1. Solder the female sockets for gold-pins in rows G and O on prototyping board (if there is no letter on yours, count the rows and check if ESP32 suits in sockets before soldering it)
2. Solder 2 JST connectors to the board in places similar to one marked on 9. The side with cutout should face closer boarder of a prototyping board.
3. Solder sensors as shown on fig:example$_o f_r outing_b ack. On the$**bottom**$o f the bard connect 5V bus and GND bus as shown on S older sen 5V). multi-meter check if there is no short circuit and a fter connecting it with power module check if there the voltages are right.$
4. Make connections between ESP32 GPIO ports and modules. On photo there is no pull-up resistors for SCL and SDA buses of I2C interface, but they are recommended to use. They should be placed according to 11.

About I2C interface:

It uses 2 signal lines: SCL (clock line) and SDA (data line). To communicate with sensor following sequence of signals should be send:

1. Start sequence (blue rectangle)
2. 7 bit address of slave device (in this case sensor address, usually given in datasheet)
3. R/W bit : responsible for marking if data will be read or written to a slave device
4. Acknowledge bit
5. 8 data bits

Last two are send till the end of transmission.

Figure 8: Multimeter

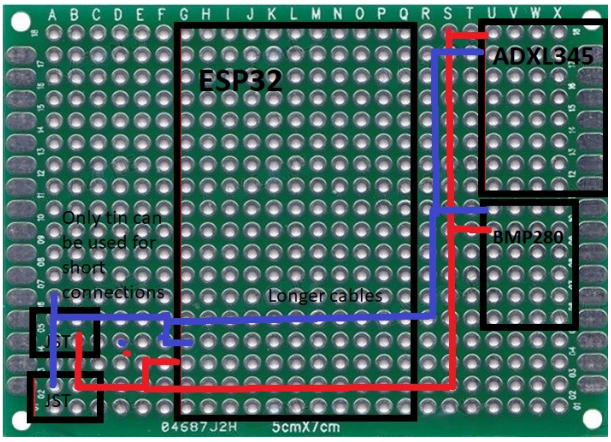


Figure 9: Example of routing power on prototyping board - please, do it on the bottom of the board.

3.5. Software and communication with sensors

3.5.1. Setting up the IDE

The ESP32 is programmed using C++ language and Arduino framework. Popular choices of IDE (integrated development environments) for this task are:

- PlatformIO - add-on for Visual Studio Code

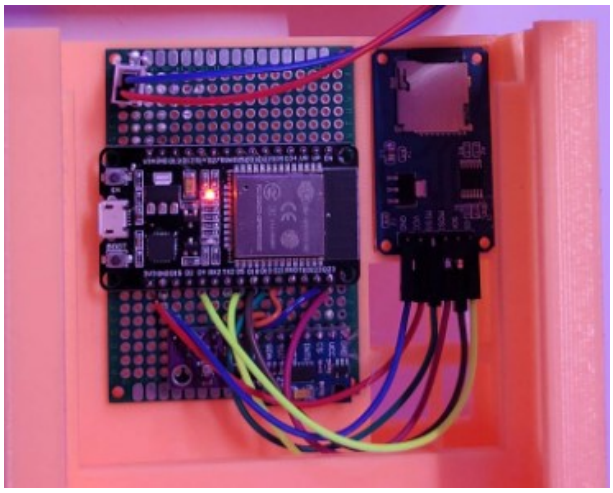


Figure 10: Example of routing signals on board - please do it on the top side

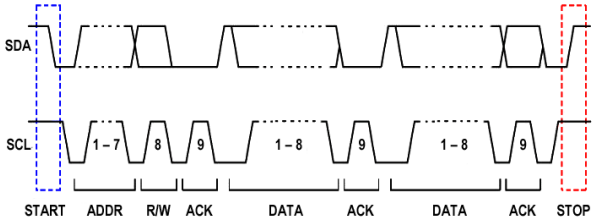


Figure 11: Signal sequences on I2C lines.

- Arduino IDE

Below setting up an ArduinoIDE will be described:

1. Download ArduinoIDE form ”https://www.arduino.cc/en/software”
2. Create new project (**File →New project**)
3. Install package with settings for ESP32 boards (**Tool →Board →Board Manager** marked blue on fig. 12)
4. After installation, choose the board (**Tool →Board →esp32** marked yellow on fig. 12)
5. Choose COM port that the board is connected to laptop (**Tool →Port** marked yellow on fig. 14)

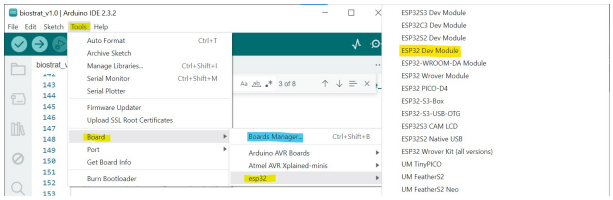


Figure 12: Choosing board.

Now connect ESP32 board to computer via USB-microUSB cable.

3.5.2. Main program

Main program includes:

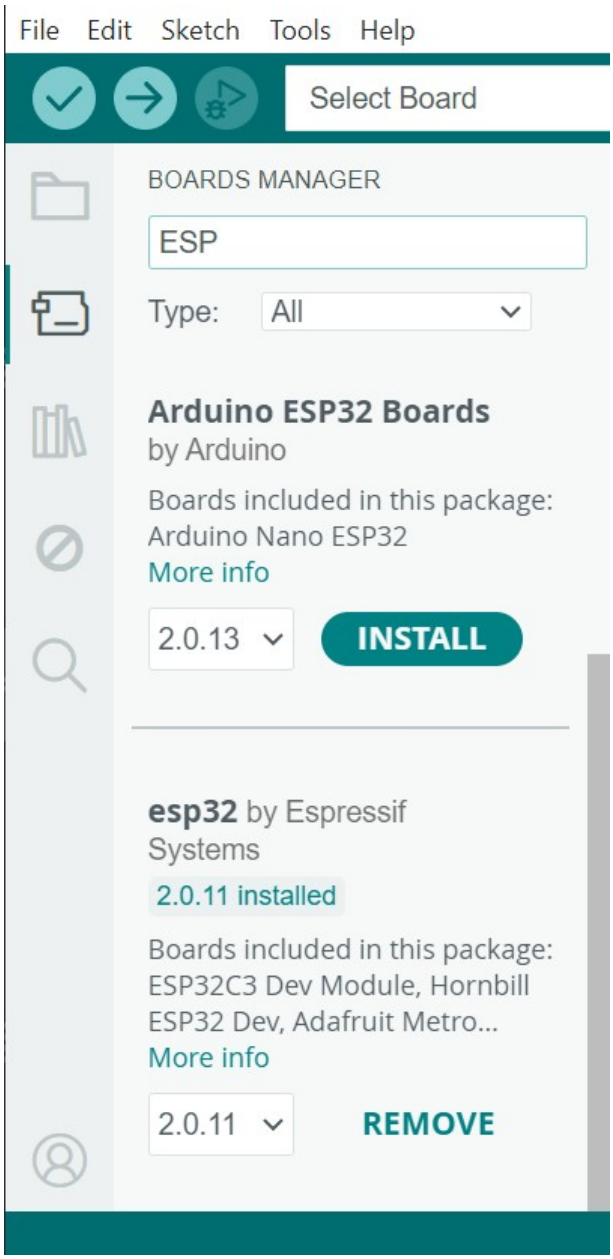


Figure 13: Choosing board.

- Declaration section - include and define directives, variable declarations and user function definitions are placed here.
- Setup loop section - start sequences are placed here.
- Main loop section - reading from sensors are placed here.
- Definition section

To communicate with slaves on software level libraray Wire.h will be used. It will manage specifics of sequences shown on fig. 7. Only thing the user have to know is slave address. In this case:

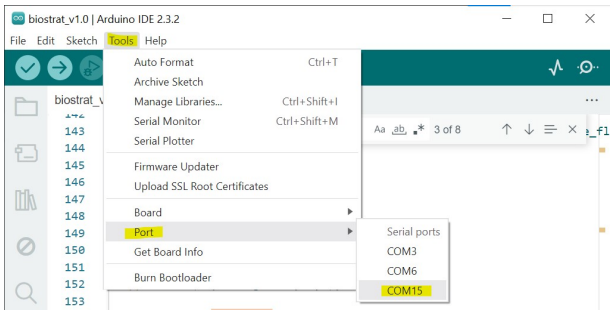


Figure 14: Choosing COM port.

- ALX345 has address 0x1D or 0x53 (0x53 can be set by giving state HIGH on SDO pin)
- BMP280 has address 0x77 or 0x76(Connecting SDO to GND results in slave address 1110110 (0x76); connection it to VDDIO results in slave address 1110111 (0x77))

If one of listed above is not working best approach is to try another. Below, using the example of ADXL385, I will demonstrate how to communicate with the sensor:

1. In declatarion section define a constant, that will stode slave address of a sensor:

Listing 1: Defining the address

```
#include <Arduino.h>
#include <Wire.h>
...
#define LED 2
#define ADXL 0x53 //accelerometer, 83 in DEC
```

2. In setup section start communication via I2C:

Listing 2: Setting up a connection

```
  void setup() {
// put your setup code here, to run once:
Serial.begin(9600);
Wire.begin(21,22);
...
}
```

3. Set the registers of accelerometers to proper configuration. First send the address of aregister in the sensor taht you want to write in, then send the data:

Listing 3: Sending configuration to sensor

```
void setup() {
...
Wire.beginTransmission(ADXL);
Wire.write(0x32); //Send the address of register that you want
                  //to write the data in the sensor
Wire.write(0x0B); //send the data that you want to write at
                  //that address, here: +- 16g range of mesurement
Wire.endTransmission();
Wire.beginTransmission(ADXL);
Wire.write(0x2D); //Power ctl
```

```
    Wire.write(0x08); //constant mesurement
    Wire.endTransmission();
    ...
    }
```

4. In the main section read data from the sensor:

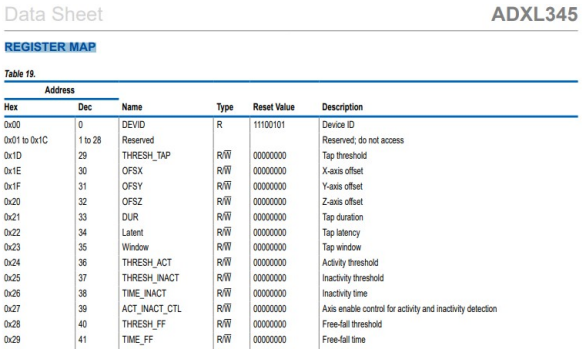
Listing 4: Reading from sensor

```
void loop() {
    static int16_t acc_raw[3];
  Wire.beginTransmission(ADXL);
  Wire.write(0x32);
  if(!Wire.endTransmission()){

    Wire.requestFrom(ADXL, 6);
    for (int i = 0; i < 6; i ++) {
      accelerations_raw[i] = Wire.read();
    }

    int16_t Xraw = accelerations_raw[0] | accelerations_raw[1] << 8;
    int16_t Yraw = accelerations_raw[2] | accelerations_raw[3] << 8;
    int16_t Zraw = accelerations_raw[4] | accelerations_raw[5] << 8;
}
```

To get the register address ant what value should be send/retreived to it one have to examin sensor datasheet and register map. Here is example of register map for ADXL345, shown at fig. 15



Data Sheet ADXL345

REGISTER MAP

Table 19.

Address					
Hex	Dec	Name	Type	Reset Value	Description
0x00	0	DEVID	R	11100101	Device ID
0x01 to 0x1C	1 to 28	Reserved			Reserved; do not access
0x1D	29	THRESH_TAP	R/W	00000000	Tap threshold
0x1E	30	OFSX	R/W	00000000	X-axis offset
0x1F	31	OFSY	R/W	00000000	Y-axis offset
0x20	32	OFSZ	R/W	00000000	Z-axis offset
0x21	33	DUR	R/W	00000000	Tap duration
0x22	34	Latent	R/W	00000000	Tap latency
0x23	35	Window	R/W	00000000	Tap window
0x24	36	THRESH_ACT	R/W	00000000	Activity threshold
0x25	37	THRESH_INACT	R/W	00000000	Inactivity threshold
0x26	38	TIME_INACT	R/W	00000000	Inactivity time
0x27	39	ACT_INACT_CTL	R/W	00000000	Axis enable control for activity and inactivity detection
0x28	40	THRESH_FF	R/W	00000000	Free-fall threshold
0x29	41	TIME_FF	R/W	00000000	Free-fall time

Figure 15: Register map of ADXL.

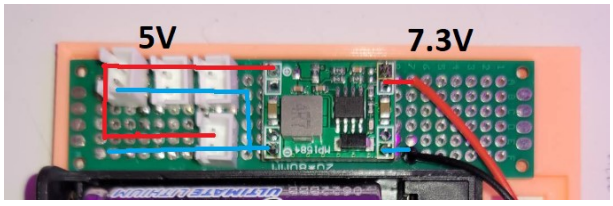


Figure 16: Register map of ADXL.

Library Wire.h and SC.h will also take care of writing/reading form SD card. The details of SPI will not be discussed here. To be able start communication with slave on SPI one have to set Chip Select line of slave to high. It this case it is connected to pin GPIO 5.

This is just a fragment of code, based on legacy code from Biostart1.0. Whole code is available at Github Repository.

3.6. Assemblay of power unit - basic version

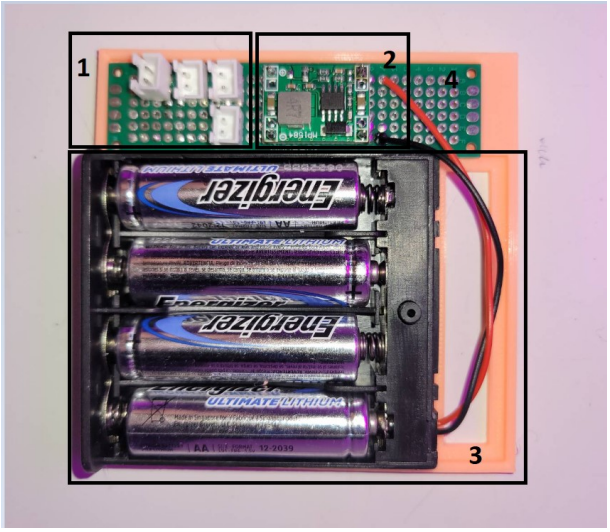


Figure 17: Power unit for a payload

Materials needed:

- min. 2 x JST XH2.54 sockets - commonly used for power distribution i hobby projects, compatible with commonly used goldpin connectors,
- 1 x Step-down converter DC-DC MP1584EN max. 3A
- 1 x Battery case for four AA batteries
- 1 x double sided prototyping board 20mm x 80 mm
- 4 x AA batteries

JST XH2.54 (no. 1 on 17) sockets are recommended, because they are commonly used for power distribution i hobby projects, compatible with popular gold-pin connectors.

Recommended batteries used in stratospheric missions, that does not have special power requirements are **Energizer Ultimate Lithium AA**. They are characterized by a long operational life, high energy density, and good performance even at low temperatures (no. 3 on 17).

Four AA batteries connected in series gives open-circuit voltage of value about 7.3V. It is too much to safely power the ESP32 and other components. This is why the DC-DC converter is used. It lowers the voltage to safe 5V, which will be distributed across the modules via JST connectors. After every connection **please check if there is no short circuit using multi-meter**. Additionally, this converter is making high tone noise when it detects short circuit.

To summarize the procedure:

1. Solder cables from battery case in similar positions as shown on a photo 17
2. Solder DC-DC converter to prototyping board using 4 gold pins. **Look at the back of converter where is side with output 5V**, this side should be faced to JST connectors.

3. Solder in JST connectrs. The side with the cutout should face DC-DC converter.
4. On the bottom side make connections for 5V and GND buses, as on 18 .
5. With multimeter check if there is no short cirtuit between 5V and GNC pins of JST connectors.
6. Power it up with switch on the bottom of battery case and check with multimeter if the voltages on JST connectors are correct (should be 5V).

The power connections are made at the **bottom side** of the prototyping board. The example of routing the connections are shown on 18.

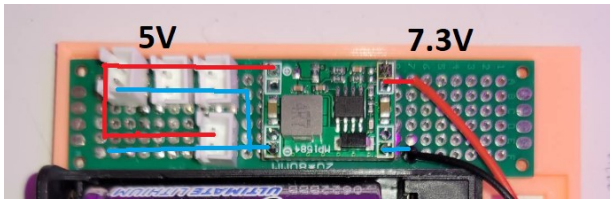


Figure 18: Routing of +5V and GND on prototyping board

4. Results

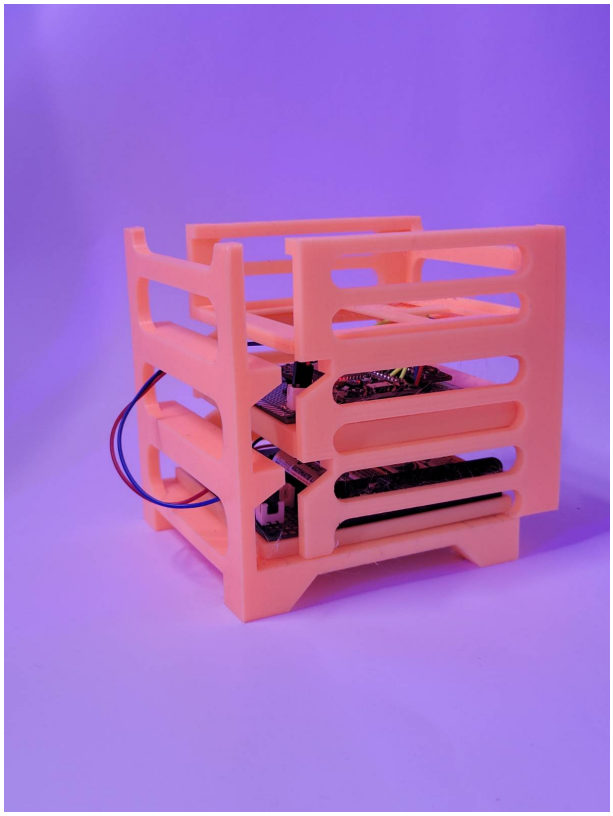


Figure 19: Final product]

The above instructions outline how to assemble a basic version of a capsule, which can serve as a platform for various experiments. With its modular design, users can flexibly adjust shelves and the number of sensors in the

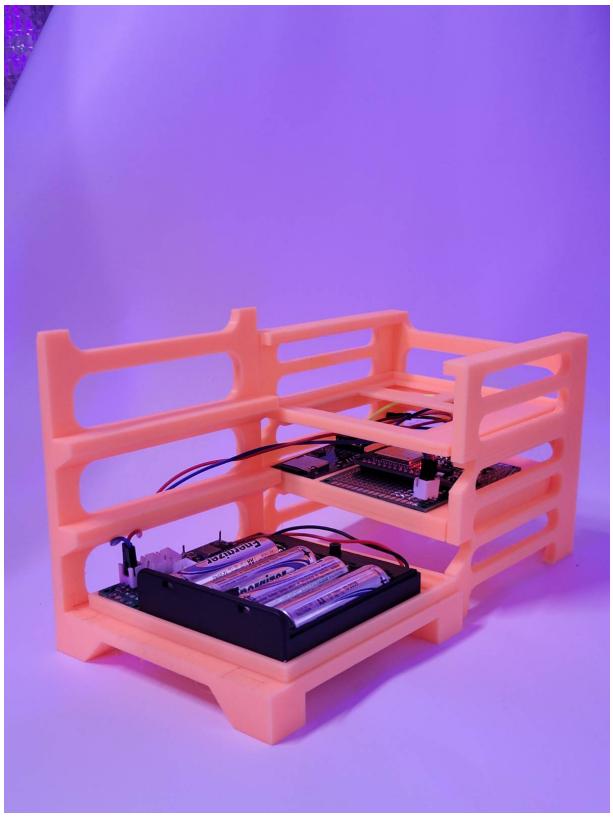


Figure 20: Angle view of assembled unit

system. This is not intended to be a comprehensive guide but rather a base that allows users to expand the system as needed. It also suggests materials that are widely available and inexpensive. Additionally, the capsule's design emphasizes ease of assembly and maintenance, making it suitable for a wide range of users, including those with limited technical expertise. Its adaptable structure enables integration with diverse experimental setups, promoting creativity and innovation in research and development projects. Moreover, the suggested materials prioritize affordability and accessibility, fostering inclusivity and collaboration within the scientific community.

5. Conclusion

During the task execution, certain decisions were made that could be optimized in subsequent iterations. For instance, reducing the thickness of the print walls can decrease the module's weight. Furthermore, transitioning part of the power supply to utilize Li-Ion batteries, known for their higher energy density, can result in a lighter power component while still ensuring sufficient energy capacity. However, such implementation necessitates deeper expertise in electronics, requiring the integration of a Battery Management System (BMS) to regulate safe charging currents and safeguard against over-discharge and overcharge conditions. Moreover, exploring alternative manufacturing methods such as injection molding could further streamline production and enhance scalability. Additionally, conducting rigorous testing and validation procedures will be crucial to ensure the reliability and safety of these modifications before widespread deployment. Finally, fostering a collaborative environment for knowledge-sharing and skill development among team members can expedite the optimization process and foster innovation. Furthermore, it is noteworthy that the project can be entirely realized with the aid of open-source tools. While the use of Fusion 360, subject to licensing constraints, is prevalent, alternatives such as FreeCAD can seamlessly substitute it, ensuring accessibility and compliance with open-source principles. Additionally, for electronic documentation, KiCad software was utilized, aligning with the project's ethos of leveraging open-source resources wherever feasible. Embracing open-source so-

lutions not only promotes transparency and collaboration but also empowers individuals and communities worldwide to participate in and contribute to the project's development.

References